cs 239 Lecture 3: Readable Software Systems **Handout 1**

Course Overview

Location	TR 2–3:50pm, Franz 1260
Instructor	Eddie Kohler, Boelter 4531c, kohler@cs.ucla.edu
Office hours	M 2–3pm or by appointment
Course Web page	http://www.cs.ucla.edu/~kohler/class/04s-readable/

Tentative schedule

1.	T 4/6	Introduction
		Servers I
2.	R 4/8	Servers II
3.	T 4/13	Servers III
4.	R 4/15	Measurement & profiling I
5.	T 4/20	Measurement & profiling II
		This week: Discuss project ideas with instructor
6.	R 4/22	Networking stacks I
7.	T 4/27	Networking stacks II
	R 4/29	**NO CLASS**
8.	T 5/4	Sensors I
		Project proposals due
9.	R 5/6	Sensors II
10.	T 5/11	OS kernels I
11.	R 5/13	OS kernels II
12.	T 5/18	Network protocols I
13.	R 5/20	Network protocols II
14.	T 5/25	Security I
15.	R 5/27	Security II
16.	T 6/1	Topics I
17.	R 6/3	Topics II
		Projects due
18.	T 6/8	Project presentations
19.	R 6/10	Project presentations

Course goals

Speedy, reliable, flexible software systems must be easy to program, so that programmers can analyze and improve the systems' performance and correctness. This seminar will look at recent systems literature, focusing on tractability, programmability, and ease of analysis: in a word, readability.

We'll concentrate on systems concerns, with occasional detours into programming language techniques.

When you have completed the course, I want you to be able to evaluate a system – a server, kernel, API, network protocol, or whatever else – on its programmability and performance. I want you to have an idea of what systems interfaces and techniques have worked, what problems systems currently face, and where programming language techniques are most needed.

Assignments

The main assignment is a course project: a significant systems effort guided by readability. One type of good project would be to take a software system important for your research and use techniques learned throughout the term to make it more "readable". The project writeup would contain both code comparisons and in-depth measurements, showing how the system improved its performance and/or reliability.

You will also be expected to read and present papers from the recent systems literature in class, and there will be occasional assignments.

Grading

25% paper presentations and discussion, 75% assignments and projects. No midterm or final exam.

Papers: Servers

These links will be made available on the course Web page.

Technical papers

Vivek S. Pai, Peter Druschel, and Willy Zwaenepoel. "Flash: An efficient and portable Web server." In *Proc. USENIX 1999*. http://www.cs.princeton.edu/~vivek/flash99/

Matt Welsh, David Culler, and Eric Brewer. "SEDA: An Architecture for Well-Conditioned, Scalable Internet Services." In *Proc. 18th SOSP*, 2001. http://www.eecs.harvard.edu/~mdw/papers/seda-sosp01.pdf

Rob von Behren, Jeremy Condit, Feng Zhou, George C. Necula, and Eric Brewer. "Capriccio: Scalable Threads for Internet Services." In *Proc. 19th SOSP*, 2003. http://capriccio.cs.berkeley.edu/pubs/capriccio-sosp-2003.pdf

Tim Brecht, David Pariag, and Louay Gammo. "accept()able Strategies for Improving Web Server Performance." To appear in *Proc. USENIX 2004*. http://www.cs.ucla.edu/~kohler/class/04s-readable/acceptable.pdf – please do not redistribute this pre-publication version.

Mor Harchol-Balter, Bianca Schroeder, Nikhil Bansal, and Mukesh Agrawal. "Size-based Scheduling to Improve Web Performance." *ACM Transactions on Computer Systems* **21**(2), May 2003, pp 207–233. http://www-2.cs.cmu.edu/~harchol/Papers/tocs257.ps

Gaurav Banga, Jeffrey C. Mogul, and Peter Druschel. "A Scalable and Explicit Event Delivery Mechanism for UNIX." In *Proc. USENIX 1999*. http://www.cs.rice.edu/~gaurav/papers/usenix99.ps

Position papers

John Ousterhout. "Why Threads are a Bad Idea (for most purposes)." Invited talk at USENIX 1996. http://www.softpanorama.org/People/Ousterhout/Threads/ or http://www.cc.gatech.edu/ccg/people/rob/software/threads/ousterhout_threads.html

Rob von Behren, Jeremy Condit, and Eric Brewer. "Why Events are a Bad Idea (for high-concurrency servers)." In *Proc. HotOS IX*. http://capriccio.cs.berkeley.edu/pubs/threads-hotos-2003.pdf

Frank Dabek, Nickolai Zeldovich, Franks Kaashoek, David Mazières, and Robert Morris. "Event-Driven Programming for Robust Software." In *Proc. 2002 SIGOPS European Workshop*. http://www.pdos.lcs.mit.edu/papers/events:sigops/

M. Frans Kaashoek, Dawson R. Engler, Gregory R. Ganger, and Deborah A. Wallach. "Server Operating Systems." In *Proc. 1996 SIGOPS European Workshop*. http://www.pdos.lcs.

mit.edu/papers/serverOS.html

Secondary references

Nickolai Zeldovich, Alexander Yip, Frank Dabek, Robert Morris, David Mazières, and Frans Kaashoek. “Multiprocessor Support for Event-Driven Programs.” In *Proc. USENIX 2003*. http://www.pdos.lcs.mit.edu/papers/asyncmp/

Vivek S. Pai, Peter Druschel, and Willy Zwaenepoel. “IO-Lite: A Unified I/O Buffering and Caching System.” *ACM Transactions on Computer Systems* **18**(1), Feb. 2000, pp 37–66. http://www.cs.princeton.edu/~vivek/iol-tocs.ps.gz

Gaurav Banga, Peter Druschel, Jeffrey C. Mogul. “Resource containers: A new facility for resource management in server systems.” In *Proc. OSDI 1999*. http://www.cs.rice.edu/~gaurav/papers/osdi99.ps